Engineering Gaze Case Study

Open Access Teaching Case Developed for the Tech for Humanity Pathways Minor

Funded by the Andrew Mellon Foundation

Developed by Dr. Kendall Giles

Background

The concept of the "engineering gaze" is inspired by the concept of the "male gaze" from feminist film theory, which was meant to summarize and critique the predominant perspective, for example, in visual media as one that objectifies women for the sexual benefit of a heterosexual male audience (Mulvey, 1989). While there are similarities between the male gaze and the engineering gaze—the engineering profession is still dominated by men—with the engineering gaze concept I focus instead on capturing the professional perspective of the engineer and their relationship to the world by looking at the technologies they design.

In order to understand the engineering gaze we must first establish what we mean by an engineer. In "American Ideologies of Science and Engineering", Edwin Layton compared the ideologies of science and engineering in the 19th and 20th centuries to find that there were distinct differences between science and engineering worldviews, yet also that a symbiotic relationship had formed between the disciplines. Whereas scientists typically viewed science as a discipline whose purpose was to create new knowledge about the world, a view championed for example by Vannevar Bush in 1945, there was a symmetric consensus that engineers were the ones who would then apply that new knowledge in the world. However, Layton, summarizing William McClellan's address to the Institute of Electrical Engineers in 1913, said that "there were actually three types of engineers: the applied scientist, the mechanic, and the designer", with McClellan preferring the latter as the "real" engineer (Layton, 1976, 696). It is this subclass of engineer, the designer—and more specifically those in the field of software engineering—that is our concern here. I am focusing on software engineering as a case study because artificial intelligence systems, a broad, and sometimes difficult to explicitly and succinctly define, collection of software and hardware technologies that can variously encompass machine learning, expert systems, neural networks, deep learning systems, computer vision, robotics, and natural language processing systems, today are mostly developed within the field of software engineering.

According to IEEE (the Institute of Electrical and Electronics Engineers), one of the largest technical professional organizations, software engineering has been defined as "the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software" ("IEEE STD 610.12-1990" 1990, 67). According to the SWEBOK v3.0, which is the latest version of the software engineering field's recognized body of knowledge and collection of best practices, it was statistician John Tukey who coined the term *software*; and *software engineering* was first used in 1968 as the title of a NATO conference in Germany (Bourque and Fairley, 2014, xvii). Foundational principles of the field are that "software engineering methods provide an organized and systematic approach to developing software for a target computer" (Bourque and Fairley, 2014, 7–9). While software system development processes generally include the steps of problem formulation, analysis, design, implementation, debugging, and testing, in practice there are a wide variety of design methods, programming languages, data structures, algorithms, models, and tools that can be used and combined in many different ways by software engineers to create the resulting system.

Thus, while there are formal and structured software development methodologies to provide a systematic approach to the development of a software solution, such as waterfall, spiral, and agile methods (ISO/IEC/IEEE 2008), actual methods used by software engineers in reality can differ based on the engineer's choices, their education background and training experience, the team's and organization's cultures, and trends within the software engineering field. From an analysis of these formalized methods and how they are used in practice, Fitzgerald noted that "different developers will not interpret and apply the same methodology in the same way; nor will the same developer apply the same methodology in the same way in different development situations. Therefore, on any development project, the methodology-in-action is uniquely enacted or instantiated by the developer" (Fitzgerald 1998, 108). Drawing on the works of Theodore Schatzki and Karin Knorr Cetina, Dittrich therefore situates software engineering as a field that has "shared social practices based on common understandings, rules, and teleoaffective structures" and which "unfolds its object [software system] and its own practice as the team proceeds in the development" (Dittrich 2016, 22–23).

Underlying these various practices and experiences is a shared ideology. One factor that influences the development of this ideology is the engineers' training, which is formally started during their engineering education in college. There is no single software engineering curriculum, though common elements of a software engineering program typically include the software design lifecycle, requirements engineering, abstraction, use cases, modeling, simulation, data analysis, programming, and visualization. Yet, at the same time a fundamental ideology inculcated across software engineering curricula is the belief that the engineer's purpose is to rationally and objectively formulate, define, scope, design, implement, and test the system that is meant to solve the identified problem.

For example, an initial step in the software engineering process is identifying and demarking the problem to be solved for the customer, which is situated in some problem domain such as medicine, law, automotive, science, security, manufacturing, or automation. A typical definition of this *problem domain* is "that part of the universe within which the problems exist" (Bray 2002, 9). End users interacting with the proposed system solution—their choices and agency—are then modeled by the engineer in order to determine the needed system functionality, which then drives the requirements specification and system design. However, the problem is underdetermined, in that the engineer also has choice and agency in deciding which requirements to specify, the criteria by which the requirements should be met, and which of the many possible designs could be implemented to meet the needed functionality. But while the choices and agency of the users of the system are explicitly represented in these specifications, the choices and agency of the engineers are not. In other words, in software engineering the problem domain is a function of the use of the resulting problem solution in some particular application environment, which exists in a part of the universe not inhabited by the software engineers themselves. The engineers exist outside the universe of the customer, users, and the problem itself, and it is from this distanced, "neutral," objective perspective that engineers are taught as a basis to rationally render system requirements, designs, and implementations.

Moreover, this distancing of the engineer from the users, problem, and final ends of the system they are designing results in the engineers' perception that they only need to focus on formulating and solving the problem they have defined as essential. In other words, the whole purpose for the engineer is the means of a system, not the ends: “This is one aspect of the ideology of technology—technological problem-solving becomes an end in and of itself, irrespective of larger considerations. Or, to put the matter more accurately, the question of means is the dominant (even sole) consideration and the question of the value of ends to which they are the means is left to take care of itself” (Graham 1999). To an engineer, value considerations about a technology only come from properties of the design of the problem solution, properties such as efficiency, performance, scale, or cost—there are no value considerations for how that design impacts society. In fact, there is a shared understanding, following the so-called Value Neutrality Thesis (Pitt 2014), that technology is neutral, and so the engineers are free of any responsibilities for downstream sociotechnical negative unintended consequences.

To consider an analogy, in science Knorr Cetina describes a purification process whereby objects from nature are improved, cleaned, denoised, summarized, and extracted when scientists transition them for use in the laboratory (Cetina, 1995). Robert Kohler documents an example of this process with fruit flies, where efforts are made to transform drosophila from a creature of nature to a creature of the lab in order to aid the scientists in their genetic mapping efforts (Kohler, 1994). This science is not performed in the world nor on the world. Rather, the science is *of* the world, performed in the lab on simplified extractions from the world—e.g., the fruit flies taken from the world, but thereafter bred for generation after generation within the lab to purposely exhibit different genetic traits. The perspective and concerns of laboratory scientists are thus of the world, not in the world. Similarly, engineering is built upon a process based on the design of problem solutions that are objectivized, abstracted, rationalized, distanced, detached, extracted, and clarified from the real world. Thus, it is these attributes and this ideology that capture the perspective and relationship of the engineer to the world and their approach to problem and solution identification, simplification, design, and implementation. This then is the engineering gaze—the engineers and the problems they solve are extracted from the world; the engineers and their practices are of the world, not in the world.

Case Study

As a specific case study and extended example of the engineering gaze, I want to trace its origin in the field of artificial intelligence to the present day, showing through a socio-institutional analysis how the people involved in its creation and development instituted an ideology and perspective that created the conditions for what some might call today’s crisis of failure in AI.

Thomas Kuhn developed the idea of a “paradigm” as the dominant framework within which scientists do their day-to-day scientific research. The paradigm for a particular scientific discipline is a shared model of research that helps scientists identify which problems are valid for research and publication, and in general includes fundamental theories, methods, and values that guide scientists’ research (Kuhn, 1970). So, just as Kuhn’s concept of a paradigm guides what scientists see in the world—scientists only observe what fits within their paradigm—I use a socio-institutional lens here to show that the way engineers are trained to view the world and how they understand their relationship to the world similarly impacts how and what they see and do in practice—how they find and define problems, as well as how they design and implement problem solutions. Thus, how and what engineers see in the world impacts their practice, reinforced by their professional societies, which in turn impacts their education programs.

Especially in the US, the 1950s in particular saw research excitement and energy trying specifically to capitalize on opportunities presented by the collision of advances in information theory (e,g., Shannon, 1948), control theory (e.g., Wiener, 2019), logic (e.g., Whitehead and Russell, 1925/1927), computer systems (e.g., Von Neumann, 1993), and theories of the mind (e.g., Gardner, 1987). From within this research context four researchers, John McCarthy, Marvin Minsky, Nathaniel Rochester, and Claude Shannon, joined together in 1956 to propose the first conference in the world[1] to work on what was coined the problem of “artificial intelligence” (McCarthy et al. 2006).

What institutional backgrounds, skills, and viewpoints were necessary for defining and working on the problem of artificial intelligence as they defined it? In the context of the “hard core” of a scientific research program (Lakatos, 1968), and broadly corresponding to the disciplines involved when conducting “normal science” within a research paradigm (Kuhn, 1970), conference proposers and participants such as Oliver Selfridge, Ray Solomonoff, Allen Newell, Herbert Simon, Trenchard More, and Alex Bernstein,[2] came from backgrounds in psychology, electrical engineering, mathematics, and physics. Other contemporaries who were not physically present at the Dartmouth conference, but were nevertheless collaborators with and influencers of those at the conference, included Alan Turing, Warren McCulloch, Walter Pitts, Arthur

[1] The conference took place at Dartmouth University.
[2] http://raysolomonoff.com/dartmouth/boxbdart/dart56ray812825who.pdf, accessed 31 March 2025.

Samuel, Donald Hebb, Dean Edmonds, Norbert Wiener, Frank Rosenblatt, and John von Neumann, all of whom had similar education and research backgrounds.
Some participants were exceptions—for example, although Herbert Simon's Ph.D was in Political Science, his contributions to AI were shaped by his research in Psychology and Logic, as demonstrated in *The Sciences of the Artificial* (Simon, 2019), among other works. Similarly, while Allen Newell achieved his Ph.D under Simon in Industrial Administration, they worked together to develop the Logic Theorist computer program and together they later earned the Turing Award,[3] a prominent award in Computation (Computer Science)---not their "home" discipline of Industrial Administration.

Of those core disciplines mentioned above, Mathematics dominated amongst these founders of AI by approximately two to one over the other fields, with Psychology, Electrical Engineering, and Physics represented in numbers comparable to each other. While Mathematics and Psychology are perhaps obvious fields to make contributions to AI, Electrical Engineering and Physics were also needed to move things forward. Theory mostly came from Mathematics and Psychology. Due to the early computer systems available at the time, implementations of their algorithms and demonstrations of early neural network models were "programmed" using analog hardware circuits as well as assembly languages such as IPL (Corporation, Kelly, and Newell, 1964), skills especially available to graduates of the Electrical Engineering and Physics programs of that era. Thus, from a socio-institutional perspective, these core disciplines dominated the worldview of the founders of AI, providing them with tools and perspectives that focused on mathematics, rational decision making, symbolic reasoning, modeling, and simulation.

As an example of how this predominant set of disciplinary worldviews shaped how AI problems were defined and solved, let us consider the proposed sub-problems of artificial intelligence that were to be the research focus of the Dartmouth conference. In the conference proposal, published on August 31, 1955, the authors viewed the universe of artificial intelligence to encompass the following sub-problems: 1) Automatic Computers, 2) How Can a Computer Be Programmed to Use a Language, 3) Neuron Nets, 4) Theory of the Size of a Calculation, 5) Machine Self-Improvement, 6) Abstractions, and, 7) Randomness and Creativity (McCarthy et al., 2006). These sub-problems involved simulating machines using automatic calculators, programming a computer to use language modeled on how humans think about and use words,

[3] "Allen Newell," https://amturing.acm.org/award_winners/newell_3167755.cfm, accessed 31 March 2025.

arranging groups of artificial neurons to form what to humans are concepts, determining the complexity of devices needed to efficiently calculate the solution to a “well-defined” problem, proposing schemes for how truly intelligent machines might perform self-improvement activities, classifying and describing machine methods for forming abstractions, and programming creative thinking through controlled randomness.

While these sub-problems were not necessarily intended for the creation of new knowledge about nature, as a pure scientist might intend, those AI researchers did use nature as an inspiration for how they idealized, simplified, abstracted, scoped, and defined problems they wished to solve, and that process then shaped the means by which the researchers could create designs for each problem’s solution.

That many of those proposed sub-problems remain open problems today belies the researchers’ confidence that they could make significant progress on them over the course of those two months at Dartmouth in the summer of 1956. Perhaps this over-confidence in the ability to solve or find solutions for those defined problems is also a root characteristic of the engineer’s worldview. As such, the perspective, ideology, and relationship to the world of the founders of AI can be seen to embody and exemplify the engineering gaze. This gaze guided the founders of AI in defining the field, deciding what problems were considered valid “AI problems,” identifying what socio-institutional backgrounds were appropriate for membership in the AI researcher community, and providing the exemplary designs, methods, and approaches that were deemed acceptable for sub-problem solutions.

We can still see AI’s engineering gaze in today’s world by similarly considering a socio-institutional perspective to analyze how AI has been solidified in higher education institutions’ training of new AI engineers. One example can be seen in what many consider the standard textbook in the field, Russell and Norvig’s *Artificial Intelligence: A Modern Approach* (2003). The first version, published in 1995, argued that Philosophy, Mathematics, Psychology, Computer Engineering, and Computational Linguistics formed the foundational disciplines of AI. To compare this list with the disciplines that formed the core of AI’s founding researchers, I first want to note that Computational Linguistics was not yet a field in and of itself in 1956, and yet some founding AI researchers were interested in exactly that, but by another name, with the computational manipulation of language as seen in sub-problem 2) “How Can a Computer Be Programmed to Use a Language.” Also, Russell and Norvig categorized people and schools of thought (reaching back to Aristotle) that focused on rational decision-making and connecting knowledge with action as philosophers, whereas I noted the Mathematics background of AI founders with similar research concentrations in Logic and rational decision-making. Finally, I will note Russell and Norvig's anachronistic inclusion of "Computer Engineering" as a foundational discipline—that discipline is actually a modern name for a discipline encompassing skills previously taught especially in Electrical Engineering—the comparable disciplinary background of a significant number of the AI founders. Indeed, the first Computer Science department in the US was not created until 1962,[4] and the first accredited Computer Engineering degree in the US was not awarded until 1971.[5]

In the textbook's 4th edition, published in 2020, the authors also highlight Economics as a foundational field, noting, for example, the economic field's later embrace of Herbert Simon's work on bounded rationality and satisficing. I have noted Simon's research background in Psychology; Russell and Norvig also included Neuroscience in the 4th edition for reasons including post-1956 research done on brain-machine interfaces. However, it is more accurate to say that the foundational AI researchers worked on neuroscience problems using technologies of the times—electro-mechanical models of the brain's neuron activities. Finally, the 2020 edition included the domains of Control Theory and Cybernetics, although early AI founders who worked on Control Theory and Cybernetics problems actually had backgrounds in Electrical Engineering and Mathematics.

We see from this overview of the field's early history that despite the evolution and semantic changes of some of the discipline names, the disciplinary perspectives of the founding AI researchers are still represented in the AI that is taught today. In fact, Mendon-Plasek describes the worldview of today's AI engineers in language that echoes our development of the engineering gaze: "the problem-framing strategies and practices of machine learning now in ascendancy were articulated, made durable, and widely circulated by researchers working on pattern recognition problems from the 1950s to the 1970s" (2021, 32). Thus, the discipline of AI, including the researcher's educational training, worldview, and perception of what problems and solution approaches are viewed as valid has remained relatively unchanged from an engineering gaze standpoint since its inception, despite the field weathering some critiques over the years. In fact, there have even been suggestions that the AI discipline actively guards its

[4] "Computer science pioneer Samuel D. Conte dies at 85," https://www.cs.purdue.edu/about/conte.html, accessed 31 March 2025.
[5] "History," https://engineering.case.edu/about/history, accessed 31 March 2025.

worldview from outside perspectives, for example as noted by McCorduck: “Accusations of clannishness have persisted since 1956, and they aren’t without foundation” (2004, 130). Critically, this closed and closeted nature of the engineering gaze in AI has caused what some might call a crisis of failure in AI today, as the AI discipline’s engineering gaze remains steadfast despite seeming inadequate for responding to and dealing with the unprecedented scale and scope of the negative unintended impacts of AI in society.

Processing Questions

1) How does engineering relate to science? How does what engineers do differ from what scientists do?

2) What is a paradigm?

3) What is a software development methodology?

4) Engineers view themselves as neutral, objective, and rational problem solvers. Is this true, and why or why not is this view problematic?

5) What academic fields form the foundation of AI?

6) Can researchers from outside AI’s foundation fields “do” AI? How might AI researchers from within AI’s foundation fields view the work of those outside those fields?

7) Compare and contrast the “male gaze” with the “engineering gaze.”

8) What are the advantages and disadvantages of researchers from outside engineering working in and on AI?

9) From an education perspective, what does it take to work in a technological field?

Thematic Reflection and Discussion

Perspective

Traditional engineering methodologies intentionally leave the human outside of the scope of the system being designed. These approaches “black box” the system and only consider the human role in terms of how the human will interface with the system, to give inputs or receive outputs. The engineer focuses on the design (the means), and not the effects of the technology once it is released into the world (the ends). The effect of this way of viewing the world, and this perspective determining what problems exist and how they can be solved, brings to mind those optical illusions we enjoyed when we were children. One of those well-known illusions, Rubin's vase (Figure 1), is a picture that can be seen in two ways—as a vase, or two human faces in profile. I'll bet you can recall seeing the Rubin's vase for the first time and being delighted when your mind clicked and you could see both the vase and the faces one after the other, switching back and forth.

Figure 1. An example of a Rubin's Vase image. Xu, Hanwen, Daiki Matsumoto, Koki Kanazawa, and Junichi Takeno. "Using a Conscious System to Construct a Model of the Rubin's Vase Phenomenon." Procedia Computer Science 88 (2016): 27-32.

The catch with the engineering gaze though is that engineers are trained to see the world in only one way: they only see the vase. The danger of this single perspective is summarized by the following quote from researcher Deborah Raji:

> *Technologists are not like doctors, looking each patient in the eye. They stand at a distance, the relationship between their judgement and system outcomes blurred by digitised abstraction, their sense of responsibility dampened by scale, the rush of agile innovation, countless undocumented judgements, and implicit feature engineering. The result is an imagined absolution of responsibility, a false narrative in which they've created an artificial system outside of anyone's control, while the human population affected by their decisions and mistakes is inappropriately erased. (Raji 2021, 59)*

So, when researchers from outside Engineering raise their concerns about the many problems caused by AI failures in society, the engineers effectively cannot see or hear them. Engineers don’t have the language or perspective to understand the world in another way—they can’t see the faces, only the vase.

Questions

1) What are some factors that shape the perspective of someone working in a particular field, such as engineering, medicine, law, or science?

2) What are some ways someone in a particular field could broaden or change their perspective for how they view the world?

3) For high risk and high impact technologies, such as AI in “self-driving” cars, how might companies, organizations, governments, and engineers mitigate the negative impacts in society of the technologies they design?

Design and Problem Solving

Researchers such as Meredith Broussard (2019), Safiya Noble (2018), Rumman Chowdhury (Rakova et al., 2021), Cathy O’Neil (2016), and Joy Buolamwini (2023) have documented many ongoing negative impacts of AI in society, characterized as a crisis of AI failure that harms people, organizations, and the environment. Their work exposes AI risks as biased, unsafe, untrustworthy, unethical, privacy-violating, and rights-violating. There have been piecemeal efforts to address those issues, resulting in research, development, and policy efforts focused on, for example, Ethical AI (Prem, 2023), Explainable AI (Dwivedi et al., 2023), Responsible AI (Agarwal and Mishra, 2021), Unbiased AI (Kartal, 2022), Safe AI (Morales-Forero, Bassetto, and Coatanea, 2023), Transparent AI (Räuker et al., 2023), and Fair AI (Correa et al., 2022). Within engineering practices such as computer science and software engineering, these efforts are not only unsystematic and disjointed, but also extremely narrow; the systems under development have their boundaries encompassing just the AI model, inputs, and outputs—a traditional engineering algorithmic design and problem-solving approach. These efforts fail to address the sociotechnical nature of AI. AI systems are designed and implemented in a social context regardless of how objective the designers may consider their own trained perspective to be. Simply abstracting the technical concerns away from the social concerns leads to inherently flawed systems.

Consider what is known as “Fair AI,” for example; a typical approach in engineering is to narrowly define “fairness” as some metric or formula that can be implemented in software code and assess the “fairness” of the resulting system. But “fair” according to whom? Fairness is a concept that has been debated in society for centuries, and what is considered “fair” varies from community to community, varies by context, and varies over time. So the chance of a programmer without any understanding of sociotechnical concerns successfully distilling such a broad, malleable, and value-laden concept into an equation when developing the AI system is virtually impossible. A well-known example arose in a system designed to suggest sentencing recommendations for judges: through this AI assessment algorithm, black defendants were likely to be falsely classified as high risk of committing future crimes at twice the rate as white defendants, while white defendants were falsely classified as low risk of committing future crimes more often than black defendants (Angwin, J. et al. 2016). Not only did the system which engineers believed they had objectively designed to treat defendants fairly end up discriminating against minority populations, but the AI system’s definition of fairness neither incorporated nor modeled the reality of users of the systems. Disparities arise in “fair AI” as subjectivity inevitably enters into all human decisions. Some judges may be biased toward following the decisions of the system, just because AI made some calculations the judge assumes must be valid; other judges may refuse to use computed recommendations—the so-called “automation bias” (Skitka et al., 2000); and still other judges may adopt the AI’s decision based on contingent or contextual factors as they determine them. Thus, even setting aside the inherently biased design that caused discriminating sentencing decisions, how can the AI system be considered “fair” when the system’s definition of fairness did not incorporate the variability in how the system would be used in its social context?

Questions

1) Why, or why not, is engineering’s perspective that technical solutions exist for social problems problematic?

2) Instead of Biased AI, Responsible AI, or Fair AI, what might a Sociotechnical AI design process look like?

Academic Disciplines and Interdisciplinarity

Especially in academia, we hear a lot about how important it is to do interdisciplinary work, and yet vastly more funding is dedicated to STEM (Science, Technology, Engineering, and Math) disciplines rather than disciplines in the Liberal Arts and Human Sciences. We have seen how most of the work done on exposing and understanding today's crisis of AI failure comes from disciplines outside engineering. Thus, if interdisciplinary work is actually important—and in the context of AI, essential—why do things seem to stay the same? Why do the disciplines, structures, and cultures on campus (or government, or industry) seem to stay siloed in their traditional organizational structures and ways of looking at the world?

When you delineate an academic discipline, such as Physics, English, Electrical and Computer Engineering, or Philosophy, you are describing boundaries that academics have constructed to contain faculty communities, each with different epistemic cultures. Biology, for example, is an academic discipline that contains faculty trying to understand the world by methods including (but not limited to) taking samples of biological organisms from their natural environment, bringing them into a laboratory, and manipulating those organisms within that controlled environment to better understand how those organisms work. Disciplines specify the objects we study (e.g., genes, deviant behavior, or classic texts) and the relations that obtain from them (mutation, criminality, canonicity). Disciplines provide criteria for defining knowledge, such as truth, significance, and impact, and methods of obtaining it, such as quantification, interpretation, and analysis (Klein 2021, 17).

Thus the disciplines are organized to teach and research certain methods, topics, techniques, and concepts, and not others. Disciplines create their own epistemic cultural boundaries, and true interdisciplinarity involves crossing those boundaries. Julie Klein uses the metaphor of critical mass to suggest that while disciplines can change over time and new disciplines can form, in order for changes to stick a tipping point must be reached: there must be enough dedicated scholars and educators, large enough research and educational programs, sufficient finances, infrastructures, and scholarly knowledge, and a self-defined identity within the area of scholarship to move transformation of a discipline forward (Klein 2021, 60). Faculty within a discipline have their particular established bodies of knowledge, research processes and methods, notions of rigor, and ways of distributing new discoveries and information, so there is often rigid resistance to any attempts by transgressing outsiders to bring in new methods or to change existing methods. So, change can happen, but it can take a long time and needs a significant mass of support to make any change lasting.

Questions

1) Why is interdisciplinarity important in designing complex technological systems?

2) Why is interdisciplinarity difficult to achieve in academia, government, or industry?

3) How might critical approaches from the Humanities add value to engineering designs or otherwise improve technological implementations in society?

Engineering Ethics

How do we broaden the historically narrow and insular engineering educational culture and dismantle the notion that engineers are uniquely objective and rational, and therefore not responsible for what their roles in the design of technical systems may create in the world outside their development silo? One example attempt is the introduction of "engineering ethics" into Engineering curricula. Ethics education has been a part of the accreditation standards for undergraduate Engineering education since 2000 (Engineering Accreditation Commission 2004). Some see it as a way of encouraging "upstream integration of social concerns into engineering practice, in the hope of developing 'heterogeneous engineers' or 'reflexive engineers'" (Frow and Calvert 2013). However, to date, such approaches have had minimal impacts. Assessments of ethics education for engineers in universities have shown poor results (Hamad et al. 2013), as well as clear misalignment of teaching objectives and outcomes (Bairaktarova and Woodcock 2015). Causes of these failures have been seen:

1) at the individual instructor level—for example from lack of clarity in appropriate ethics pedagogy methods to use, lack of sufficient course content, and lack of expertise,

2) at the institutional level—for example from low emphasis on the importance of ethics, lack of systematic approaches for ethics training across the institution, low emphasis on ethics from accrediting organizations, and superficial implementations of ethics education in universities just to meet minimal accreditation compliance standards, and

3) at the level of engineering culture—which echoes concerns of the engineering gaze we have developed in this case study—including, for example, engineering training that valorizes "the technical over the social", conflicting education paradigms between Engineering and the Humanities, and an Engineering disciplinary culture fostering a hierarchy of "hard" sciences and knowledge above the "soft" humanities and social sciences (Martin, Conlon, and Bowe 2021).

Yet, while methods used to teach ethics to engineers in general may serve as a starting point, clearly there must be modifications to update ethical instruction for software and AI engineers. For example, Narayanan and Vallor point out that pedagogical content for engineering ethics typically comes from domains such as Mechanical, Civil, and Electrical, but not Software Engineering, much less AI (Narayanan and Vallor, 2014). In addition to the difficulties and failures in incorporating ethics education into undergraduate engineering programs, as we have discussed in this paper, the engineering gaze needs to expand its perspective beyond just a perfunctory attention to ethics. In order to address the crisis of failure in AI, engineers also need training to build awareness and responsibility regarding issues with bias, transparency, safety, security, fairness, privacy, interpretability, accountability, and risk which their designs may perpetuate. Fundamentally, we require mitigation steps to help transform the engineering gaze from one that is distanced from the world to one that consciously inhabits and engages with the world. The AI Engineering field's lack of critical introspection has been noted by a number of scholars (e.g., Downey and Zuiderent-Jerak, 2016; Felt et al., 2013; Robbins, 2007) as a lack of reflexivity in engineers, and from our analysis of the engineering gaze, this lack of reflexivity seems to apply in particular in the comparatively emergent field of AI.

Questions

1) How might engineers better understand their role in helping create and maintain the crisis of AI failure?

2) Part of engineering ethics training is to help provide the engineers a language or perspective that can be used to understand issues and communicate with those outside their discipline, what some have characterized as a translation problem. What is the role of the Humanities, if any, in this translation problem and solution?

3) Beyond engineering ethics, what other concepts, topics, or areas–what languages or perspectives–are missing from engineering education that are critical for designing complex and risky technologies that are then implemented in society?

Bibliography

Agarwal, Sray, and Shashin Mishra. 2021. *Responsible AI*. Springer.

Angwin, J., Larson, J., Mattu, S., and Kirchner, L. 2016. "Machine Bias. There's Software Used Across the Country to Predict Future Criminals. And It's Biased Against Blacks." *Propublica*.

Bairaktarova, Diana, and Anna Woodcock. 2015. “Engineering Ethics Education: Aligning Practice and Outcomes.” *IEEE Communications Magazine* 53 (11): 18–22.

Bourque, Pierre, and R.E. Fairley. 2014. “SWEBOK V3. 0.” *IEEE Computer Society*.

Bray, Ian K. 2002. *An Introduction to Requirements Engineering*. Pearson Education.

Broussard, Meredith. 2019. *Artificial Unintelligence: How Computers Misunderstand the World*. Illustrated edition. The MIT Press.

Buolamwini, Joy. 2023. *Unmasking AI: My Mission to Protect What Is Human in a World of Machines*. Random House.

Bush, Vannevar. 1945. “Science, The Endless Frontier: A Report to the President.”

Correa, Ramon, Mahtab Shaan, Hari Trivedi, Bhavik Patel, Leo Anthony G Celi, Judy W Gichoya, and Imon Banerjee. 2022. “A Systematic Review of ‘Fair’ AI Model Development for Image Classification and Prediction.” *Journal of Medical and Biological Engineering* 42 (6): 816–27.

Dittrich, Yvonne. 2016. “What Does It Mean to Use a Method? Towards a Practice Theory for Software Engineering.” *Information and Software Technology* 70: 220–31.

Downey, Gary Lee, and Teun Zuiderent-Jerak. 2016. “Making and Doing: Engagement and Reflexive Learning in STS.” *Handbook of Science and Technology Studies*, 223–50.

Dwivedi, Rudresh, Devam Dave, Het Naik, Smiti Singhal, Rana Omer, Pankesh Patel, Bin Qian, et al. 2023. “Explainable AI (XAI): Core Ideas, Techniques, and Solutions.” *ACM Computing Surveys* 55 (9): 1–33.

Engineering Accreditation Commission. 2004. “Accreditation Board for Engineering and Technology (ABET).” Baltimore, MD: Criteria for Accrediting Engineering Programs.

Felt, Ulrike, Daniel Barben, Alan Irwin, Pierre-Benoit Joly, Arie Rip, Andy Stirling, and Tereza Stöckelová. 2013. “Science in Society: Caring for Our Futures in Turbulent Times Science Policy Briefing No. 50.” Strasbourg: European Science Foundation.

Fitzgerald, Brian. 1998. “An Empirically-Grounded Framework for the Information Systems Development Process.” *ICIS 1998 Proceedings* 10.

Frow, Emma, and Jane Calvert. 2013. “‘Can Simple Biological Systems Be Built from Standardized Interchangeable Parts?’Negotiating Biology and Engineering in a Synthetic Biology Competition.” *Engineering Studies* 5 (1): 42–58.

Gardner, Howard. 1987. *The Mind’s New Science: A History of the Cognitive Revolution*. Basic Books.

Graham, Gordon 1949 July 15-. 1999. *The Internet : A Philosophical Inquiry*. London ; Routledge.

Hamad, Jehan Abu, Maram Hasanain, Mahmoud Abdulwahed, and Rashid Al-Ammari. 2013. “Ethics in Engineering Education: A Literature Review.” In *2013 IEEE Frontiers in Education Conference (FIE)*, 1554–60. https://doi.org/10.1109/FIE.2013.6685099.

“IEEE STD 610.12-1990.” 1990. *IEEE Standard Glossary of Software Engineering Terminology*.

ISO/IEC/IEEE. 2008. “Standard for Systems and Software Engineering-Software Life Cycle Processes.” *IEEE STD*, 12207–008.

Kartal, Elif. 2022. “A Comprehensive Study on Bias in Artificial Intelligence Systems: Biased or Unbiased AI, That’s the Question!” *International Journal of Intelligent Information Technologies (IJIIT)* 18 (1): 1–23.

Kelly, Hugh S. and Allen Newell (The Rand Corporation). 1964. *Information Processing Language-V Manual*. Prentice-Hall.

Klein, Julie Thompson. 2021. *Beyond Interdisciplinarity: Boundary Work, Communication, and Collaboration*. Oxford University Press.

Knorr Cetina, Karin. 1995. “Laboratory Studies: The Cultural Approach to the Study of Science.” *Handbook of Science and Technology Studies*, 140–67.

Kohler, Robert E. 1994. *Lords of the Fly: Drosophila Genetics and the Experimental Life*. University of Chicago Press.

Kuhn, Thomas S. 1970. *The Structure of Scientific Revolutions*. University of Chicago Press.
Lakatos, Imre. 1968. “Criticism and the Methodology of Scientific Research Programmes.” In *Proceedings of the Aristotelian Society*, 69:149–86. JSTOR.

Layton, Edwin T. 1976. “American Ideologies of Science and Engineering.” *Technology and Culture* 17 (4): 688–701.

Martin, Diana Adela, Eddie Conlon, and Brian Bowe. 2021. “A Multi-level Review of Engineering Ethics Education: Towards a Socio-technical Orientation of Engineering Education for Ethics.” *Science and Engineering Ethics* 27 (5). https://doi.org/10.1007/s11948-021-00333-6.

McCarthy, John, Marvin L Minsky, Nathaniel Rochester, and Claude E Shannon. 2006. “A Proposal for the Dartmouth Summer Research Project on Artificial Intelligence, August 31, 1955.” *AI Magazine* 27 (4): 12–12.

McCorduck, Pamela. 2004. *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence*. CRC Press.

Mendon-Plasek, Aaron. 2021. “Mechanized Significance and Machine Learning: Why It Became Thinkable and Preferable to Teach Machines to Judge the World.” *The Cultural Life of Machine Learning: An Incursion into Critical AI Studies*, 31–78.

Morales-Forero, Andres, Samuel Bassetto, and Eric Coatanea. 2023. “Toward Safe AI.” *AI & SOCIETY* 38 (2): 685–96.

Mulvey, Laura. 1989. "Visual Pleasure and Narrative Cinema." In *Visual and Other Pleasures*, 14–26. Springer.

Narayanan, Arvind, and Shannon Vallor. 2014. "Why Software Engineering Courses Should Include Ethics Coverage." *Communications of the ACM* 57 (3): 23–25.

Noble, Safiya Umoja. 2018. *Algorithms of Oppression: How Search Engines Reinforce Racism*. Illustrated edition. New York: NYU Press.

O'Neil, Cathy. 2016. *Weapons of Math Destruction: How Big Data Increases Inequality and Threatens Democracy*. Broadway Books.

Pitt, Joseph C. 2014. "'Guns Don't Kill, People Kill'; Values In and/or Around Technologies." In *The Moral Status of Technical Artefacts*, 89–101. Springer.

Prem, Erich. 2023. "From Ethical AI Frameworks to Tools: A Review of Approaches." *AI and Ethics* 3 (3): 699–716.

Raji, Deborah. 2021. "The Bodies Underneath the Rubble." In *Fake AI*, edited by Frederike Kaltheuner, 53–61. Meatspace Press.

Rakova, Bogdana, Jingying Yang, Henriette Cramer, and Rumman Chowdhury. 2021. "Where Responsible AI Meets Reality: Practitioner Perspectives on Enablers for Shifting Organizational Practices." *Proceedings of the ACM on Human-Computer Interaction* 5 (CSCW1): 1–23.

Räuker, Tilman, Anson Ho, Stephen Casper, and Dylan Hadfield-Menell. 2023. "Toward Transparent Ai: A Survey on Interpreting the Inner Structures of Deep Neural Networks." In *2023 Ieee Conference on Secure and Trustworthy Machine Learning (Satml)*, 464–83. IEEE.

Robbins, Peter T. 2007. "The Reflexive Engineer: Perceptions of Integrated Development." *Journal of International Development: The Journal of the Development Studies Association* 19 (1): 99–110.

Russell, Stuart, and Peter Norvig. 2003. *Artificial Intelligence: A Modern Approach*. Prentice Hall Upper Saddle River, New Jersey, USA.

Shannon, Claude E. 1948. "A Mathematical Theory of Communication." *The Bell System Technical Journal* 27 (3): 379–423.

Simon, Herbert A. 2019. *The Sciences of the Artificial*. MIT Press.

Skitka, Linda J, Kathleen L Mosier, Mark Burdick, and Bonnie Rosenblatt. 2000. "Automation Bias and Errors: Are Crews Better Than Individuals?" *The International Journal of Aviation Psychology* 10 (1): 85–97.

Von Neumann, John. 1993. "First Draft of a Report on the EDVAC." *IEEE Annals of the History of Computing* 15 (4): 27–75.

Whitehead, Alfred North, 1861-1947., and Russell 1872-1970. 1925/1927. *Principia Mathematica*. 2d ed. Cambridge [England]: The University Press.

Wiener, Norbert. 2019. *Cybernetics or Control and Communication in the Animal and the Machine*. MIT press.